Deterministic one-way Turing machines with sublinear space bounds

Martin Kutrib, Julien Provillard, György Vaszil, Matthias Wendlandt

DETERMINISTIC ONE-WAY TURING MACHINES WITH SUBLINEAR SPACE BOUNDS

Martin Kutrib[(B)] Julien Provillard[(B)]
György Vaszil[(A)] Matthias Wendlandt[(B)]

[(B)]Institut für Informatik, Universität Giessen,
Arndtstr. 2, 35392 Giessen, Germany
{kutrib,provillard,matthias.wendlandt}@informatik.uni-giessen.de

[(A)]Department of Computer Science, University of Debrecen,
4028 Debrecen, Kassai út 26, Hungary
vaszil.gyorgy@inf.unideb.hu

Abstract
Deterministic one-way Turing machines with sublinear space bounds are systematically studied. We distinguish among the notions of strong, weak, and restricted space bounds. The latter is motivated by the study of P automata. The space available on the work tape depends on the number of input symbols read so far, instead of the entire input. The class of functions space constructible by such machines is investigated, and it is shown that every function f that is space constructible by a deterministic two-way Turing machine, is space constructible by a strongly f space-bounded deterministic one-way Turing machine as well. We prove that the restricted mode coincides with the strong mode for space constructible functions. It turns out that the strong mode is computationally less powerful than the weak mode, for any sublinear space constructible function. Moreover, the known infinite, dense, and strict hierarchy of strong space complexity classes is shown also for the weak mode by Kolmogorov complexity arguments. Finally, closure properties under Boolean operations are obtained for weak space bounds. These properties are different from the known properties of strong space bounds.

1. Introduction

The approach to measure the complexity of systems by their space requirements as opposed to time originates from [8, 11]. In these seminal papers from the early days of automata and complexity theory, two different models of Turing machines are considered, the offline and

[(A)]The author acknowledges the support of the Hungarian Scientific Research Fund, "OTKA", grant no. K75952, and the European Union through the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project which is co-financed by the European Social Fund.

online machines. Each model is deterministic and has a read-only input tape and an infinite work tape. The offline machines may read their input two-way while the online machines are not allowed to move the input head to the left. Moreover, both types of machines are assumed to halt on every input. The results from [11] have been generalized and complemented in [7] where, in particular, the halting assumption is removed, and nondeterministic computations are introduced. In the terminology of [7] the (generalized) Turing machine models are called two-way and one-way machines.

The space is bounded by the condition that, given a function f, the Turing machine uses at most $f(n)$ cells of the work tape on *every* input of length n. Since the machine has to obey the space bound for every input, it nowadays is called the *strong* space bound. One result from [11] which inspired vivid research activities till these days, is the minimal space bound for accepting non-trivial, that is, non-regular languages. For deterministic one-way machines this bound is given by the logarithm. In the meantime this threshold has been studied for several other Turing machine models. A comprehensive survey including new results is [10].

Concerning the one-way machines studied here, apart from log space-bounded computations, to our knowledge most of the known results have, in fact, been published in [11] and [7]. However, one-way machines with the least possible space bound for non-trivial computations, the logarithm, attracted and still attract great attention. For further references and results we refer again to [10]. The monograph [12] mainly deals with two-way Turing machines, whose threshold for non-trivial computations is $\log\log$ space. In [5] one-way log space *reductions* are considered. In general, log space-bounded one-way machines retain only little information about the input on the work tape. However, the reductions have been shown to be powerful enough to give complete sets for the complexity classes L, NL, P, and NP (see also [1, 3, 6]).

Here we study deterministic one-way Turing machines with space bounds. If the bounding function is below the logarithm only regular languages are accepted. On the other hand, if the bound is at least linear, the restriction to one-way input head motion is not serious. In this case, the one-way machine can first copy the whole input onto the work tape and, subsequently, can simulate a given two-way machine with the same space bound by only using the work tape. So, we are mainly interested in space bounds in the range between the logarithm and the identity. For constructible strong space bounds the existence of an infinite, dense, and strict space hierarchy has been shown by diagonalization in [11]. In [7] again the strong mode is considered. It is proved that any space-bounded deterministic one-way machine can be modified to be always halting. Moreover, nondeterminism leads to strictly more powerful devices of this type. Concerning closure properties, the closure under Boolean operations has been obtained for all space bounds in question.

Here we also consider so-called *weak* space bounds, where the space is bounded by the condition that, given a function f, the Turing machine uses at most $f(n)$ cells of the work tape on *every accepted* input of length n. So the machine may violate the space bound for rejecting computations. It is worth mentioning that both modes coincide for two-way Turing machines with constructible space bounds. One of our results is the separation of strong and weak classes. Motivated by the study of certain types of P automata, in [4] the idea came up to bound the space available dependent on the number of input symbols read so far, instead of the entire

input in advance. So, the so-called restricted mode has been introduced for log space-bounded computations. For the *restricted mode*, the space is bounded by the condition that, given a function f, the Turing machine uses at most $f(i)$ cells of the work tape in any configuration with input head at position i, occurring in any *accepting* computation. Here it turns out that the classes of languages accepted by restricted and strongly f space-bounded machines coincide, for any space constructible function f. However, in Section 2 we first present basic notions and technical details as well as an introductory example. Functions that are space constructible by the devices in question are the objects of Section 3. It is shown that every function f which is space constructible by a deterministic two-way Turing machine, is space constructible by a strongly f space-bounded deterministic one-way Turing machine as well. So, the family of such functions is very rich. By Kolmogorov complexity and incompressibility arguments the known infinite, dense, and strict hierarchy of strong space complexity classes is shown also for the weak classes in Section 5. Finally, in Section 6 closure properties under Boolean operations are obtained for weak space bounds. These properties are different from the known properties of strong space bounds.

2. Preliminaries

We denote the non-negative integers $\{0, 1, 2, \dots\}$ by $\mathbb{N}$. The *empty word* is denoted by λ and the *reversal* of a word w by w^R. For the *length* of w we write $|w|$. We use $\subseteq$ for *inclusions* and $\subset$ for *strict inclusions*. A function $f : \mathbb{N} \to \mathbb{N}$ is said to be *increasing* if $m < n$ implies $f(m) \leq f(n)$. If the latter inequality is $f(m) < f(n)$ the function is *strictly increasing*. The *inverse* of an increasing function $f : \mathbb{N} \to \mathbb{N}$ is defined as $f^{-1}(n) = \min\{\, m \in \mathbb{N} \mid f(m) \geq n \,\}$. As usual we define the set of functions that grow strictly less than f by

$$o(f) = \{\, g : \mathbb{N} \to \mathbb{N} \mid \lim_{n\to\infty} \frac{g(n)}{f(n)} = 0 \,\}.$$

In terms of orders of magnitude, f is an upper bound of the set

$$O(f) = \{\, g : \mathbb{N} \to \mathbb{N} \mid \exists\, n_0, c \in \mathbb{N} : \forall\, n \geq n_0 : g(n) \leq c \cdot f(n) \,\}.$$

Conversely, f is a lower bound of the set $\Omega(f) = \{\, g : \mathbb{N} \to \mathbb{N} \mid f \in O(g) \,\}$, and $\Theta(f)$ is defined to be $O(f) \cap \Omega(f)$.

A deterministic one-way Turing machine consists of a one-way read-only input tape whose inscription is the input word in between two endmarkers (we provide two endmarkers in order to have a definition consistent with two-way machines), an initially blank semi-infinite two-way work tape with a left endmarker, and a finite-state control. At the outset of a computation the Turing machine is in the designated initial state, and the heads of both tapes scan the left endmarkers. Dependent on the current state and the currently scanned symbols on both tapes the Turing machine changes its state, rewrites the current symbol on the work tape, and moves the heads independently. The machines have no extra output tape but the states are partitioned into accepting and rejecting states.

Definition 2.1 *A* deterministic one-way Turing machine (abbreviated as 1DTM) *is a system* $M = \langle S, \Gamma, \Sigma, \rhd, \lhd, \delta, s_0, F\rangle$, *where*

1. S *is the finite set of* internal states,
2. Γ *is the finite set of* tape symbols, *containing the* blank *symbol* $\sqcup$,
3. Σ *is the finite set of* input symbols,
4. $\rhd \notin \Gamma \cup \Sigma$ *is the* left *and* $\lhd \notin \Gamma \cup \Sigma$ *is the* right endmarker,
5. $s_0 \in S$ *is the* initial state,
6. $F \subseteq S$ *is the set of* accepting states, *and*
7. $\delta : S \times (\Sigma \cup \{\rhd, \lhd\}) \times (\Gamma \cup \{\rhd\}) \to S \times \{0, 1\} \times (\Gamma \cup \{\rhd\}) \times \{-1, 0, 1\}$ *is the partial transition function, where* -1 *means to move the head one square to the left,* 0 *means to keep the head on the current square, and* 1 *means to move one square to the right. Whenever* $(s', d_1, a', d_2) = \delta(s, \lhd, a)$ *is defined, then* $d_1 = 0$. *Whenever* $(s', d_1, a', d_2) = \delta(s, a, \rhd)$ *is defined, then* $d_2 \neq -1$ *and* $a' = \rhd$.

Let the work tape be semi-infinite to the right. Its cells are numbered from left to right beginning with cell 0. The left endmarker appears initially in cell 0. A *configuration* of a 1DTM $M = \langle S, \Gamma, \Sigma, \rhd, \lhd, \delta, s_0, F\rangle$ is a quintuple (s, w, h_1, β, h_2), where $s \in S$ is the current state, $w \in \Sigma^*$ is the input, $h_1 \in \{0, 1, \ldots, |w| + 1\}$ is the current head position on the input tape, $\beta : \mathbb{N} \to \Gamma \cup \{\rhd\}$ is a function that maps the tape cells of the work tape to their current contents, and $h_2 \geq 0$ is the current head position on the work tape. If the head position h_1 is 0, then the head is located on the left endmarker, if it satisfies $1 \leq h_1 \leq |w|$, the head is located at the h_ith letter of w, and if it is $|w| + 1$, then the head is located on the right endmarker.

The *initial configuration* for input w is set to $(s_0, w, 0, \beta, 0)$, where β maps cell 0 to the left endmarker and all the other cells to the blank symbol. During the course of its computation, M runs through a sequence of configurations. One step from a configuration to its successor configuration is denoted by $\vdash$. Let $(s, a_1a_2 \cdots a_n, h_1, \beta, h_2)$ be a configuration and $x = \rhd$ if $h_1 = 0$, $x = \lhd$ if $h_1 = |w| + 1$, and $x = a_{h_1}$ otherwise. Let $\delta(s, x, \beta(h_2)) = (s', d_1, y, d_2)$. Then

$$(s, w, h_1, \beta, h_2) \vdash (s', w, h_1 + d_1, \beta', h_2 + d_2),$$

where $\beta'(h_2) = y$ and $\beta'(m) = \beta(m)$, for $m \neq h_2$.

A Turing machine *halts* if the transition function is undefined for the current configuration. An input word w is *accepted* if the machine halts at some time in an accepting state, otherwise it is *rejected*. The *language accepted* by M is $L(M) = \{\, w \in \Sigma^* \mid w \text{ is accepted by } M \,\}$.

Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function. The 1DTM M is said to be *weakly f space-bounded* or of *weak space complexity* f, if $h_2 \leq f(|w|)$ for any configuration (s, w, h_1, β, h_2) in any *accepting* computation. That is, apart from the left endmarker, M scans at most $f(|w|)$ cells on its work tape in any accepting computation. Rejecting computations need not to obey the space bound. It is understood that we always mean $\max\{0, \lceil f \rceil\}$, in particular if we use a function that does not map to natural numbers (such as the logarithm). The 0 ensures that we have a bound for the cases where f is undefined (such as for $\log(0)$).

The idea of strongly bounded space is that the space available on the work tape depends on the number of input symbols read so far, instead of on the entire input. More precisely, the 1DTM M is said to be *strongly f space-bounded* or of *strong space complexity* f, if $h_2 \leq f(h_1)$ for *any accessible* configuration (s, w, h_1, β, h_2). That is, apart from the left endmarker, M scans at most $f(h_1)$ cells on its work tape in any computation. Loosely speaking, one can say the work tape grows while reading the input.

It is worth justifying the different modes of space bounded computations. Often, the strong mode is defined to complement the weak mode as follows. A 1DTM is said to be strongly f space-bounded, if $h_2 \leq f(|w|)$ for *any accessible* configuration (s, w, h_1, β, h_2). However, this definition is equivalent with the definition above. If $h_2 \leq f(h_1)$ for any accessible configuration then, clearly, we also have $h_2 \leq f(|w| + 1)$ (due to the endmarker). By linear compression of the work tape, the condition $h_2 \leq f(|w|)$ can always be satisfied. Conversely, let $h_2 \leq f(|w|)$ for any accessible configuration (s, w, h_1, β, h_2), and assume $f(h_1) < h_2 \leq f(|w|)$. Then we factorize the input $w = uv$ so that $|u| = h_1$ and consider a computation on input u. Since the devices are deterministic the computation reaches configuration (s, u, h_1, β, h_2). So, we have $h_2 > f(h_1) = f(|u|)$ which violates the space bound. We conclude, $h_2 \leq f(h_1)$.

Following the idea that the space available on the work tape depends on the number of input symbols read so far, in connection with P automata in [4] the so-called restricted mode has been considered for log space-bounded computations. A 1DTM is said to be *restricted f space-bounded*, if $h_2 \leq f(|h_1|)$ for any configuration (s, w, h_1, β, h_2) in any *accepting* computation. It will be shown in Theorem 3.4 that the classes of languages accepted by restricted and strongly f space-bounded 1DTM coincide, for any space constructible function f. We defer the theorem to the presentation of space constructible functions.

Clearly, if the bounding function $f(n)$ is at least linear, that is, it belongs to $\Omega(n)$, then the restriction to one-way input head motion as well as the restriction to strong space bounds are not serious. This is due to the fact, that in an initial scan the machine can copy the whole input onto the work tape and, subsequently, can simulate a given space-bounded two-way machine (for which strong and weak mode coincide) by only using the work tape. On the other hand, one-way machines with a sublogarithmic, that is, $o(\log)$ space bound accept regular languages only (see, for example, [10]). So in the following we consider mainly machines with sublinear space bounds above the logarithm.

The class of all languages which are accepted by weakly f space-bounded 1DTM is denoted by $\mathsf{1WDSPACE}(f)$. For strong space bounds we use the notation $\mathsf{1SDSPACE}(f)$.

Example 2.2 *The language* $L = \{\, a^{k^2} ww \mid k \geq 1, w \in \{b, c\}^k \,\}$ *is accepted by a strongly* $\sqrt{n}$ *space-bounded* 1DTM M.

The construction of M utilizes the fact that the sum of the first n odd positive integers is the nth square number.

The 1DTM M *uses three tracks on the work tape. On the first and second track two counters c_1 and c_2 are maintained, while the third track is used to mark the available space as well as for the main computation on input w. If the first input symbol is b, or c, or there is no input*

symbol at all, the input is rejected immediately. If it is an a, counters c_1 and c_2 are set to the second odd number 3*. Additionally, the first cell is marked on the third track.*

Now M reads the a's in phases. During one phase, counter c_2 is successively decremented. If it is 0*, the next odd number of a's has been read. In this case, counter c_1 is increased by two (to store the next odd number), and its content is copied to counter c_2. Moreover, since the position of the input head is a square number, one more tape cell is available. This cell is marked which finishes the phase.*

In this way, being on the i^2th symbol of the input, i cells on the work tape are marked and no unmarked cell has been visited. The counters take log *space, so they take less space as available.*

If the first b or c appears in the input after a phase has started, the input is rejected since the number of a's is not a square number. Otherwise, the following input symbols from $\{b, c\}$ are read and stored on the third track until the marked space is filled. In this way, exactly k symbols are stored. Finally, the remaining input is matched with the content of the third track. If this is successful, the input is accepted, otherwise rejected. □

3. Space Constructibility

In order to deal with space-bounded computations, in many cases "well-behaved" bounding functions are required. Usually the notion "well-behaved" is concretized in terms of constructibility of the functions with respect to the device in question.

Here, a function $f : \mathbb{N} \to \mathbb{N}$ is said to be *space-constructible* if it is increasing and there exists a Turing machine which, on every input a^n, visits exactly $f(n)$ cells on the work tape, not including cell 0 with the left endmarker.

Space constructible functions have been defined already in the first paper that seriously considered space-bounded computations [11]. It is well known that all "usually considered" functions above log are space constructible by two-way Turing machines, while no non-constant increasing function in $o(\log)$ is space constructible [12]. So, the family of such functions is very rich. More details can be found for example in [2, 13].

However, here we are interested in one-way machines. In order to provide a wide base of bounding functions, next we turn to translate the constructibility for two-way Turing machines to the devices in question. To this end, we first show that both models are computationally equivalent for unary languages.

Theorem 3.1 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function of order $\Omega(\log(n))$ and M be a strongly f space-bounded deterministic two-way Turing machine accepting a unary language. Then a strongly f space-bounded* 1DTM *M' can effectively be constructed so that $L(M) = L(M')$.*

Proof. Given a strongly f space-bounded deterministic two-way Turing machine M with a

unary input alphabet, M' is constructed to work in two phases.

In the first phase, M' counts the length of the input. To this end, it maintains a binary counter on the work tape, where the least significant bit is in cell 1. Whenever an input symbol has been read, the counter is increased by moving the work tape head from position 0 to the right until all carry-overs are processed. Then the head is moved back to position 0. In this way, after reading the ith input symbol, M' uses at most $\log(i)+1$ cells on the work tape. With the tacit understanding that we always can compress the work tape content linearly by increasing the tape alphabet and/or storing a constant number of symbols in the finite control, and the precondition $f \in \Omega(\log(n))$ we obtain that, so far, M' is strongly f space-bounded. Moreover, when the whole input has been read, M' has the same space available as M.

In the second phase, M' simulates the computation of M only on its work tape. Since the input is unary, to this end it suffices to store the current position of M's input head in addition to the current content of M's work tape on its own work tape. Clearly this can be realized within the given space bound. One step of M is simulated by M' as follows. First M' marks the current position of its work tape head. Then it checks whether the input head of M reads an endmarker or an input symbol. This can be done by inspecting whether its current position is 0 or $|w|+1$ or in between. For the second check it suffices to compare the current position with the length of the input counted during the first phase. Now M' moves the head back to the previously marked position on its work tape, clears the marking, and simulates one transition of M. Subsequently, it moves its head accordingly, marks the new position, and adjusts the counter that stores the current position of M's input head. Finally, it returns to the marked position and clears the marking.

Altogether the simulation takes no more space than M needs, which concludes the construction of M'. □

Now the previous theorem is applied to show that any function f that is space constructible by a deterministic two-way Turing machine is also space constructible by a strongly f space-bounded 1DTM.

Theorem 3.2 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function that is space constructible by a deterministic two-way Turing machine. Then f is space constructible by a strongly f space-bounded* 1DTM*.*

Proof. As mentioned before it is well known that no increasing function in $o(\log)$ is space constructible by deterministic two-way Turing machines [12]. So, f is of order $\Omega(\log(n))$.

Since f is space constructible by a deterministic two-way Turing machine, it is space constructible by a deterministic strongly f space-bounded two-way Turing machine by definition. Therefore, Theorem 3.1 is applicable, and we obtain a strongly f space-bounded 1DTM M' that simulates M. Moreover, the proof of Theorem 3.1 reveals that the machine M' constructed visits $\Theta(\log)$ but no more than f work tape cells in its first phase and, in addition, exactly as many cells as M in its second phase. So, since M space constructs f, M' does so either. □

Example 3.3 *The function* $\log^2$ *is space constructible by a* 1DTM M*.*

The 1DTM M *uses four tracks on its work tape. On the first three tracks three counters* c_1, c_2, *and* c_3 *are maintained, while the fourth track is used to mark the tape cells visited.*

Counter c_1 *counts the length of the input and, thus, takes* $\log(i)$ *space, where* i *is the number of symbols already read. Whenever a carry over to a blank cell is processed, that is, the length of* c_1 *increases,* $\log(i)$ *and, thus,* $\log^2(i)$ *change. In order to recalculate* $\log^2(i)$*, now the content* $\log(i)$ *of* c_1 *is copied to* c_2*, and* c_3 *is set to* 0*. Subsequently, counter* c_2 *is successively decreased, where after each decrementation the content of* c_1 *is added to* c_3*. So, when* c_2 *becomes* 0*, the content of* c_3 *is* $\log^2(i)$*. Finally, the content of track four is cleared, and the counter* c_3 *is successively decreased, where after each decrementation one blank cell on track four is marked. When* c_3 *becomes* 0*, the next input symbol is read. If* M *reaches the end of the input tape, the machine stops.* □

So, all members of the rich family of functions space constructible by general Turing machines can be used for our purposes. Just to mention prominent members, $\log(n)$, n^k for $k \geq 0$, 2^n, $n!$, and p_n, where p_n is the nth prime number, do belong to the family. Example 2.2 reveals that $\sqrt{n}$ is also a member. Moreover, the class is closed under several operations. If f_1 and f_2 are space constructible, then so are $f_1(n) + f_2(n)$, $f_1(n) \cdot f_2(n)$, $2^{f_1(n)}$, $f_1(n)^{f_2(n)}$, and many others.

We conclude the section by showing that the classes of languages accepted by restricted and strongly space-bounded 1DTM coincide.

Lemma 3.4 *Let* $f : \mathbb{N} \to \mathbb{N}$ *be a function space constructible by a* 1DTM *and* M *be a restricted* f *space-bounded* 1DTM*. Then a strongly* f *space-bounded* 1DTM M' *accepting* $L(M)$ *can effectively be constructed.*

Proof. The definition of the restricted mode says that in accepting computations M already behaves as a strongly space-bounded machine. So, we can try to detect if $h_2 > f(h_1)$ in a configuration (s, w, h_1, β, h_2), which violates the strong space bound. In this situation the computation cannot continue accepting and the input can safely be rejected. The result is a strongly space-bounded machine.

In order to detect if $h_2 > f(h_1)$, M' is constructed as follows. Basically, it simulates M and a 1DTM C that space constructs f. In addition, it maintains a counter on the work tape. The counter is increased whenever the input head is moved to the right.

More precisely, M' simulates M step by step. After every step the simulation is interrupted. Whenever the input head is moved to the right, say to position i, the counter is increased. Then the space constructor C is simulated on input of length i given by the counter. This can be done on extra tracks of the work tape without destroying the counter. During this sub-simulation every cell on the work tape visited is marked. So, exactly the leftmost $f(i)$ cells are marked. Now, it is easy for M' to detect whether the work tape head would be moved out of the marked area, that is, whether $h_2 > f(h_1)$. □

Since by definition, every strongly space-bounded 1DTM is also restricted space-bounded, by Lemma 3.4 the next corollary follows.

Corollary 3.5 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function space constructible by a* 1DTM. *The classes of languages accepted by restricted and strongly f space-bounded* 1DTM *coincide.*

4. Strong versus Weak Space Bounds

This section is devoted to the natural question, whether or not strong space bounds are less powerful than weak space bounds for deterministic one-way Turing machines. The following example provides us with witness languages for the main result of this section.

Example 4.1 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function that is space constructible by a* 1DTM. *The language*

$$L_f = \{\, wc^k w \mid w \in \{a,b\}^*, k \geq 1, |w| \leq f(k) \,\}$$

is accepted by a weakly f space-bounded 1DTM *M.*

In a first phase, M reads the prefix w and stores it onto the work tape. When the first c appears in the input, it moves the work tape head to the left endmarker and starts to simulate the 1DTM *that space constructs the function f. The simulation is done on input c^k. During the space construction all tape cells visited are marked on an extra track. After the simulation is completed, M checks whether the input prefix w is stored only on marked tape cells. If not we have $|w| > f(k)$ and M rejects. Otherwise we have $|w| \leq f(k) \leq f(wc^k w)$ since f is space constructible and, therefore, increasing. So, M obeys the space bound f. Finally, M compares the remaining input suffix with the prefix stored on the work tape. If both match, M accepts, otherwise it rejects.* □

Next, we turn to show that, for any sublinear function f, the language L_f cannot be accepted by any strongly f space-bounded 1DTM.

Theorem 4.2 *Let $f : \mathbb{N} \to \mathbb{N}$ be a sublinear function space constructible by a* 1DTM. *Then the class* $\mathsf{1SDSPACE}(f)$ *is properly included in* $\mathsf{1WDSPACE}(f)$.

Proof. The inclusion follows immediately from the definition. So, it suffices to prove its properness. By Example 4.1 the language L_f is accepted by a weakly f space-bounded 1DTM. It remains to be shown, that L_f is not accepted by any strongly f space-bounded 1DTM.

For that we use Kolmogorov complexity and an incompressibility argument. General information on Kolmogorov complexity and the incompressibility method can be found, for example, in [9]. Let $w \in \{a,b\}^+$ be an arbitrary binary string. The Kolmogorov complexity $C(w)$ of w is defined to be the minimal size of a program describing w. The following key argument for the incompressibility method is well known. There are binary strings w of *any* length so that $|w| \leq C(w)$.

Assume in contrast to the assertion that L_f is accepted by some strongly f space-bounded 1DTM $M = \langle S, \Gamma, \Sigma, \rhd, \lhd, \delta, s_0, F \rangle$. We choose a word $z = wc^k w \in L_f$ so that $C(w) \geq |w|$, consider an accepting computation on z, and show that w can be compressed. To this end let

$c_1 = (s, z, 1, \beta, h_2)$ be the configuration of M when the input head has left the left endmarker, and $c_2 = (s', z, |w| + 1, \beta', h_2')$ be the configuration of M, when the input head has reached the first symbol c. Omitting the second component z, each of these configurations can be encoded with

$$\begin{aligned} & O(\log(S) + \log(|w|) + \log(\Gamma^{f(|w|)}) + \log(f(|w|))) \\ = {} & O(\log(|w|) + f(|w|)\log(\Gamma) + \log(f(|w|))) \\ = {} & O(\log(|w|) + f(|w|)) \end{aligned}$$

bits. Since f is space constructible by a 1DTM and, thus, of order $\Omega(\log(n))$, we obtain $O(f(|w|))$ bits.

Knowing M and the two configurations without the second component z, we can reconstruct w as follows. For each candidate string x of length $|w|$, the Turing machine M is simulated beginning in configuration $(s, x, 1, \beta, h_2)$. If the simulation reaches configuration $(s', x, |x| + 1, \beta', h_2')$ with a move of the input head, we know $x = w$. If in this situation $x \neq w$, there would be a computation beginning in the initial configuration $(s_0, xc^k w, 0, \beta, 0)$ reaching configurations $(s, xc^k w, 1, \beta, h_2)$ and $(s', xc^k w, x + 1, \beta', h_2')$. Now the computation continues exactly as for c_2 and accepts. However, the input $xc^k w$ has to be rejected.

So, we have $C(w) = C(M) + O(f(|w|)) + c = O(f(|w|))$, for a positive constant c which gives the size of the program reconstructing w. Since f is sublinear, we conclude $C(w) < |w|$, for w long enough. This is a contradiction to the fact that $C(w) \geq |w|$. □

It is worth mentioning that the ternary alphabet used for the witness languages of Theorem 4.2 can easily be reduced to a binary alphabet by standard encodings.

5. Hierarchies

In [11] an infinite dense space hierarchy of strongly space-bounded 1DTM is shown by diagonalization. Here we derive such hierarchies for strongly and weakly space bounded 1DTM, where the classes are separated by Kolmogorov complexity arguments. In this way, also witness languages separating the classes are obtained.

Example 5.1 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function that is space constructible by a* 1DTM. *The language*

$$L_f' = \{\, c^k w \$ w \mid w \in \{a, b\}^*, k \geq 1, |w| \leq f(k + |w|) \,\}$$

is accepted by a strongly f space-bounded 1DTM *M.*

As one task, M simulates the 1DTM *that space constructs the function f. The simulation is done on input $c^k w$. During the space construction all tape cells visited are marked on an extra track. While there are symbols c in the input, M only simulates the constructor. When the infix w is read, M additionally stores it onto the work tape whereby only marked cells are used. If $|w| \leq f(k + |w|)$ there is space enough to store w symbol by symbol. Otherwise w is too long*

and $f(k+|w|) < |w|$. In this case the input is rejected. We have $|w| \leq f(k+|w|) \leq f(|c^k w\$w|)$ since f is space constructible and, therefore, increasing. So, M obeys the space bound f. Finally, after reading the \$, M compares the remaining input suffix with the word stored on the work tape. If both match, M accepts, otherwise it rejects. □

Next, we turn to show that, for any sublinear function f_1, the language L'_{f_1} cannot be accepted by any strongly f_2 space-bounded 1DTM, if $f_2 \in o(f_1)$. As in a previous proof, we use Kolmogorov complexity arguments.

Lemma 5.2 *Let $f_1 : \mathbb{N} \to \mathbb{N}$ and $f_2 : \mathbb{N} \to \mathbb{N}$ be two sublinear functions space constructible by* 1DTM *with $f_2 \in o(f_1)$. Then the language L'_{f_1} is not accepted by any weakly f_2 space-bounded* 1DTM.

Proof. Assume in contrast to the assertion that the language L'_{f_1} is accepted by some weakly f_2 space-bounded 1DTM $M = \langle S, \Gamma, \Sigma, \rhd, \lhd, \delta, s_0, F\rangle$. We choose a word $z = c^k w\$w \in L'_{f_1}$ so that $C(w) \geq |w|$ and $f_1(k+|w|) \leq |w| < f_1(k+|w|+1)$, consider an accepting computation on z, and show that z can be compressed. To this end, let $c_1 = (s, z, k+1, \beta, h_2)$ be the configuration of M when the input has been moved from the last symbol c, and $c_2 = (s', z, k+|w|+1, \beta', h'_2)$ be the configuration of M, when the input head has reached the \$. Omitting the second component z, each of these configurations can be encoded with

$$\begin{aligned} & O(\log(S) + \log(k+|w|) + \log(\Gamma^{f_2(k+|w|)}) + \log(f_2(k+|w|))) \\ = {} & O(\log(k+|w|) + f_2(k+|w|)\log(\Gamma) + \log(f_2(k+|w|))) \\ = {} & O(\log(k+|w|) + f_2(k+|w|)) \end{aligned}$$

bits. Since f_2 is space constructible by a 1DTM and, therefore, of order $\Omega(\log(n))$, we obtain $O(f_2(k+|w|))$ bits.

Knowing M and the two configurations without the second component z, we can reconstruct w as follows. For each candidate string x of length $|w|$, the Turing machine M is simulated beginning in configuration $(s, x, 1, \beta, h_2)$. If the simulation reaches configuration $(s', x, |x|+1, \beta', h'_2)$ with a move of the input head, we know $x = w$.

So, we have $C(w) = C(M) + O(f_2(k+|w|)) + c = O(f_2(k+|w|))$, for a positive constant c. Since $f_2 \in o(f_1)$, we obtain $C(w) \in o(f_1(k+|w|)) = o(|w|)$ and, hence, $C(w) < |w|$, for w long enough. This is a contradiction to the fact that $C(w) \geq |w|$. □

Example 5.1 and Lemma 5.2 are useful sources for witness languages. As side effect infinite dense space hierarchies for weakly as well as strongly space-bounded 1DTM follow.

Theorem 5.3 *Let $f_1 : \mathbb{N} \to \mathbb{N}$ and $f_2 : \mathbb{N} \to \mathbb{N}$ be two functions space constructible by* 1DTM *with $f_2 \in o(f_1)$. Then the class* 1SDSPACE(f_2) *is properly included in* 1SDSPACE(f_1)*, and the class* 1WDSPACE(f_2) *is properly included in* 1WDSPACE(f_1).

Proof. The inclusions are evident. If f_1 is at least linear, their properness follows from the well-known hierarchy for two-way Turing machines. Otherwise Example 5.1 provides a language L'_{f_1} accepted by a strongly f_1 space-bounded 1DTM, and Lemma 5.2 shows that L'_{f_1} is not accepted by any weakly f_2 space-bounded 1DTM. So, both inclusions claimed are proper. □

6. Closure Properties

Besides the fact that closure properties can shed some light on the structure of a complexity class they may be used as powerful reduction tools in order to simplify proofs or constructions. Closure under certain operations indicates a certain robustness of the language families, while non-closure properties may serve, for example, as a valuable basis for extensions. Here we consider the Boolean operations for weakly (and strongly) space-bounded 1DTM. In [7] the closure of $\mathsf{1SDSPACE}(f)$ under Boolean operations has been shown, where f is space constructible by a 1DTM. In order to obtain the results, the Turing machines are made to be halting on every input. For the sake of completeness we present the next theorem.

Theorem 6.1 ([7]) *Let $f : \mathbb{N} \to \mathbb{N}$ be a function space constructible by a* 1DTM. *Then the class $\mathsf{1SDSPACE}(f)$ is effectively closed under union, intersection, and complementation.*

Next we consider weakly space-bounded 1DTM. It turns out, that the closure properties are considerably different. We start with one of the few positive properties.

Theorem 6.2 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function. Then the class $\mathsf{1WDSPACE}(f)$ is effectively closed under intersection.*

Proof. Let $M = \langle S, \Gamma, \Sigma, \rhd, \lhd, \delta, s_0, F\rangle$ and $M' = \langle S', \Gamma', \Sigma', \rhd, \lhd, \delta', s'_0, F'\rangle$ are two weakly f space-bounded 1DTM. The effective construction of another weakly f space-bounded 1DTM $M'' = \langle S'', \Gamma'', \Sigma'', \rhd, \lhd, \delta'', s''_0, F''\rangle$ accepting the language $L(M) \cap L(M')$ is as follows.

Basically, the idea is to simulate both machines given. Since the input can be read one-way only, these simulations cannot be done sequentially, but have to be done in parallel. So, the state set S'' is defined as Cartesian product $S \times S'$ and s''_0 is set to (s_0, s'_0).

Both machines may modify their work tapes in completely different ways. In order to cope with this situation, the work tape of M'' uses two tracks. In addition, the current head position of M is marked by an h and the current head position of M' by an h'. Therefore, Γ'' is defined to be $\Gamma \times \Gamma' \times \{\sqcup, h, h', hh'\}$.

The last problem to overcome is the possibly different head movement on the input. It may happen that one machine moves its head to the right while the other keeps its head on the current cell. In such situations, M'' simulates a step of the the machine that keeps its head stationary but does not simulate the other machine. Clearly, M'' simulates the steps of both machines in parallel, if both heads are stationary or both move to the right.

Machine M'' rejects when one of the machines simulated rejects, and it continues to simulate the other machine when one of the machines accepts. The set of accepting states is defined as $F'' = \{\, (s, s') \mid s \in F \text{ and } s' \in F' \,\}$. So, if M'' accepts, both machines M and M' accept and, thus, obey the space bound. This in turn implies that M'' obeys the space bound as well.

The details of the the construction of δ'' are straightforward. □

Next we turn to union.

Theorem 6.3 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function space constructible by a* 1DTM. *Then the class* $\mathsf{1WDSPACE}(f)$ *is not closed under union.*

Proof. We use the language

$$L_f = \{\, wc^k w \mid w \in \{a,b\}^*, k \geq 1, |w| \leq f(k) \,\}$$

of Example 4.1 and the regular language $R = \{a,b\}^*$ as witnesses. Both are accepted by weakly f space-bounded 1DTM. Assume contrarily the union $L_f \cup R$ is accepted by some weakly f space-bounded 1DTM M as well. On input prefixes of the form $\{a,b\}^*$ machine M works as in strong mode. That is, it uses no more than $f(i)$ space on the work tape if the input head is at position i. The reason is that every such prefix has to be accepted, in which case M has to obey the space bound f. In this way, the regular language R is used to 'fool' the machine M.

Now, from M an equivalent strongly f space-bounded 1DTM M' is constructed as follows. On input prefixes of the form $\{a,b\}^*$, machine M' simulates M directly. If the whole input is of this form, M accepts. This preserves the strong mode. Otherwise, when the first c appears in the input, M' freezes its current work tape content and sets up a counter on an extra track. This counter is increased for every symbol c read. Since the counter takes at most $\log(k) \leq \log(|wc^k|) \leq f(|wc^k|)$ space, again the strong mode is preserved.

Next, M' simulates a space constructor for f on input c^k with the help of the counter, thereby marking $f(k)$ tape cells. This can be done without destroying the counter. Since $|w| \leq f(k)$, the next task of M' is to read and store the suffix w on the work tape, whereby no further space is used. If the suffix is too long, M' rejects. Altogether, M' reads its whole input in strong mode.

Finally, M' continues to simulate M with the frozen work tape content obeying the space bound f, whereby the input is provided by the counter and the suffix w stored on the work tape.

The class $\mathsf{1SDSPACE}(f)$ is closed under intersection and, in particular, under intersection with regular sets. So, $L(M') \cap \{a,b\}^* c^+ \{a,b\}^* = L_f$ is accepted by a strongly f space-bounded 1DTM as well. This is a contradiction, since in the proof of Theorem 4.2 it has been shown that L_f does not belong to the class $\mathsf{1SDSPACE}$. □

Now the non-closure under complementation follows immediately from the closure under intersection and non-closure under union. If $\mathsf{1WDSPACE}(f)$ were closed under complementation it would be closed under union.

Theorem 6.4 *Let $f : \mathbb{N} \to \mathbb{N}$ be a function space constructible by a* 1DTM. *Then the class* $\mathsf{1WDSPACE}(f)$ *is not closed under complementation.*

References

[1] E. ALLENDER, Isomorphisms and 1-L Reductions. *J. Comput. System Sci.* 36 (1988), 336–350.

[2] J. L. BALCÁZAR, J. DÍAZ, J. GABARRÓ, *Structural Complexity I*. Springer, Berlin, 1988.

[3] H.-J. BURTSCHICK, A. HOENE, The Degree Structure of 1-L Reductions. In: *Mathematical Foundations of Computer Science (MFCS 1992)*. LNCS 629, Springer, 1992, 153–161.

[4] E. CSUHAJ-VARJÚ, O. H. IBARRA, G. VASZIL, On the Computational Complexity of P Automata. *Natural Computing* 5 (2006), 109–126.

[5] J. HARTMANIS, N. IMMERMAN, S. R. MAHANEY, One-Way Log-Tape Reductions. In: *Foundations of Computer Science (FOCS 1978)*. IEEE Computer Society, 1978, 65–72.

[6] J. HARTMANIS, S. R. MAHANEY, Languages Simultaneously Complete for One-Way and Two-Way Log-Tape Automata. *SIAM J. Comput.* 10 (1981), 383–390.

[7] J. E. HOPCROFT, J. D. ULLMAN, Some Results on Tape-Bounded Turing Machines. *J. ACM* 16 (1969), 168–177.

[8] P. M. LEWIS II, R. E. STEARNS, J. HARTMANIS, Memory Bounds for Recognition of Context-Free and Context-Sensitive Languages. In: *Symposium on Switching Circuit Theory and Logical Design*. IEEE Computer Society, 1965, 191–202.

[9] M. LI, P. M. B. VITÁNYI, *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 1993.

[10] C. MEREGHETTI, Testing the Descriptional Power of Small Turing Machines on Nonregular Language Acceptance. *Int. J. Found. Comput. Sci.* 19 (2008), 827–843.

[11] R. E. STEARNS, J. HARTMANIS, P. M. LEWIS II, Hierarchies of memory limited computations. In: *Symposium on Switching Circuit Theory and Logical Design*. IEEE Computer Society, 1965, 179–190.

[12] A. SZEPIETOWSKI, *Turing Machines with Sublogarithmic Space*. LNCS 843, Springer, Berlin, 1994.

[13] K. WAGNER, G. WECHSUNG, *Computational Complexity*. Reidel, Dordrecht, 1986.